00:36:40 Exit Lab Complete Lab

Installing a Kafka Cluster and Creating a Topic

1 hour 15 minutes duration Practitioner Rate this lab

VIDEOS GUIDE

Installing a Kafka Cluster and Creating a Topic

Introduction

Apache Kafka is known for many things, one of them is the ability to process over 1 million messages per second. The ease of decoupling components makes it appealing to developers as well. In this hands-on lab, you will build a custom Kafka cluster and create a topic. A topic consists of many partitions of messages and is the fundamental unit used in Kafka.

Solution

1. Begin by logging in to the lab servers using the credentials provided on the hands-on lab page:

```
ssh cloud_user@PUBLIC_IP_ADDRESS
```

Download the Kafka Binaries

Note: Each step will need to be completed on all three lab servers.

1. Use `wget` to download the tar file from the mirror:

```
wget
http://mirror.cogentco.com/pub/apache/kafka/2.6.2/kafka_2.12-
2.6.2.tgz
```

2. Extract the tar file and move it into the `/opt` directory:

```
tar -xvf kafka_2.12-2.6.2.tgz
```

```
sudo mv kafka_2.12-2.6.2 /opt/kafka
```

3. Change to the `kafka` directory and list the contents:

```
cd /opt/kafka
```

```
ls
```

Install Java and Disable RAM Swap

Note: Each step will need to be completed on all three lab servers.

1. Use the following command to install the latest Java Developer Kit (JDK):

   ```
   sudo apt install -y default-jdk
   ```

2. Use the following command to verify that Java has been installed:

   ```
   java -version
   ```

3. Use the following command to disable RAM swap:

   ```
   swapoff -a
   ```

4. Use the following command to comment out swap in the `/etc/fstab` file:

   ```
   sudo sed -i '/ swap / s/^/#/' /etc/fstab
   ```

Create a New Directory for Kafka and Zookeeper

Note: Each step will need to be completed on all three lab servers.

1. Use the following command to create a new directory for Kafka message logs:

   ```
   sudo mkdir -p /data/kafka
   ```

2. Use the following command to create a snapshot directory for Zookeeper:

   ```
   sudo mkdir -p /data/zookeeper
   ```

3. Change ownership of those directories to allow the user `cloud_user` control:

   ```
   sudo chown -R cloud_user:cloud_user /data/kafka
   ```

   ```
   sudo chown -R cloud_user:cloud_user /data/zookeeper
   ```

Specify an ID for Each Zookeeper Server

On Server #1:

1. Use the following command to create a file in `/data/zookeeper` (on server #1) called `myid` with the contents "1" to specify Zookeeper server #1:

```
echo "1" > /data/zookeeper/myid
```

On Server #2:

2. Use the following command to create a file in `/data/zookeeper` (on server #2) called `myid` with the contents "2" to specify Zookeeper server #2:

   ```
   echo "2" > /data/zookeeper/myid
   ```

On Server #3:

3. Use the following command to create a file in `/data/zookeeper` (on server #3) called `myid` with the contents "3" to specify Zookeeper server #3:

   ```
   echo "3" > /data/zookeeper/myid
   ```

Modify the Kafka and Zookeeper Configuration Files

Note: Each step will need to be completed on all three lab servers.

1. Use the following command to remove the existing `server.properties` file (in the `config` directory) and create a new `server.properties` file:

   ```
   rm config/server.properties
   ```

   ```
   vim config/server.properties
   ```

2. Copy and paste the following into the contents of the `server.properties` file and change the `broker.id` and the `advertised.listeners`:

   ```
   # change this for each broker
   broker.id=[broker_number]
   # change this to the hostname of each broker
   advertised.listeners=PLAINTEXT://[hostname]:9092
   # The ability to delete topics
   delete.topic.enable=true
   # Where logs are stored
   log.dirs=/data/kafka
   # default number of partitions
   num.partitions=8
   # default replica count based on the number of brokers
   default.replication.factor=3
   # to protect yourself against broker failure
   min.insync.replicas=2
   # logs will be deleted after how many hours
   ```

```
log.retention.hours=168
# size of the log files
log.segment.bytes=1073741824
# check to see if any data needs to be deleted
log.retention.check.interval.ms=300000
# location of all zookeeper instances and kafka directory
zookeeper.connect=zookeeper1:2181,zookeeper2:2181,zookeeper3:2181/l

# timeout for connecting with zookeeper
zookeeper.connection.timeout.ms=6000
# automatically create topics
auto.create.topics.enable=true
```

Note: Be sure to replace `[broker_number]` with the server number. Example: For Server #3, replace `[broker_number]` with "3".

Be sure to replace `[hostname]` with the Kafka server number. Example: For Server #3, replace `[hostname]` with "kafka3".

Save and close the file once the necessary changes have been made.

3. Use the following command to remove the existing `zookeeper.properties` file (in the `config` directory) and create a new `zookeeper.properties` file:

```
rm config/zookeeper.properties
```

```
vim config/zookeeper.properties
```

4. Copy and paste the following into the contents of the `zookeeper.properties` file (don't change this file):

```
# the directory where the snapshot is stored.
dataDir=/data/zookeeper
# the port at which the clients will connect
clientPort=2181
# setting number of connections to unlimited
maxClientCnxns=0
# keeps a heartbeat of zookeeper in milliseconds
tickTime=2000
# time for initial synchronization
initLimit=10
# how many ticks can pass before timeout
syncLimit=5
# define servers ip and internal ports to zookeeper
```

```
server.1=zookeeper1:2888:3888
server.2=zookeeper2:2888:3888
server.3=zookeeper3:2888:3888
```

Create the Kafka and Zookeeper Service

Note: Each step will need to be completed on all three lab servers.

1. Create the file `/etc/init.d/zookeeper` on each server and paste in the following contents:

```
sudo vim /etc/init.d/zookeeper

#!/bin/bash
#/etc/init.d/zookeeper
DAEMON_PATH=/opt/kafka/bin
DAEMON_NAME=zookeeper
# Check that networking is up.
#[ ${NETWORKING} = "no" ] && exit 0

PATH=$PATH:$DAEMON_PATH

case "$1" in
  start)
        # Start daemon.
        pid=`ps ax | grep -i 'org.apache.zookeeper' | grep -v
grep | awk '{print $1}'`
        if [ -n "$pid" ]
          then
            echo "Zookeeper is already running";
        else
          echo "Starting $DAEMON_NAME";
          $DAEMON_PATH/zookeeper-server-start.sh -daemon
/opt/kafka/config/zookeeper.properties
        fi
        ;;
  stop)
        echo "Shutting down $DAEMON_NAME";
        $DAEMON_PATH/zookeeper-server-stop.sh
        ;;
  restart)
        $0 stop
        sleep 2
        $0 start
        ;;
  status)
        pid=`ps ax | grep -i 'org.apache.zookeeper' | grep -v grep | awk '{print $1}'`
        if [ -n "$pid" ]
          then
          echo "Zookeeper is Running as PID: $pid"
        else
          echo "Zookeeper is not Running"
        fi
        ;;
  *)
        echo "Usage: $0 {start|stop|restart|status}"
        exit 1
esac

exit 0
```

2. Change the file to executable, change ownership, install, and start the service:

```
sudo chmod +x /etc/init.d/zookeeper

sudo chown root:root /etc/init.d/zookeeper

sudo update-rc.d zookeeper defaults

sudo service zookeeper start

sudo service zookeeper status
```

3. Create the file `/etc/init.d/kafka` on each server and paste in the following contents:

```
#!/bin/bash
#/etc/init.d/kafka
DAEMON_PATH=/opt/kafka/bin
DAEMON_NAME=kafka
# Check that networking is up.
#[ ${NETWORKING} = "no" ] && exit 0

PATH=$PATH:$DAEMON_PATH

# See how we were called.
```

```
case "$1" in
  start)
        # Start daemon.
        pid=`ps ax | grep -i 'kafka.Kafka' | grep -v grep | awk
'{print $1}'`
        if [ -n "$pid" ]
          then
            echo "Kafka is already running"
        else
          echo "Starting $DAEMON_NAME"
          $DAEMON_PATH/kafka-server-start.sh -daemon
/opt/kafka/config/server.properties
        fi
        ;;
  stop)
        echo "Shutting down $DAEMON_NAME"
        $DAEMON_PATH/kafka-server-stop.sh
        ;;
  restart)
        $0 stop
        sleep 2
        $0 start
        ;;
  status)
        pid=`ps ax | grep -i 'kafka.Kafka' | grep -v grep | awk
'{print $1}'`
        if [ -n "$pid" ]
          then
          echo "Kafka is Running as PID: $pid"
        else
          echo "Kafka is not Running"
        fi
        ;;
  *)
        echo "Usage: $0 {start|stop|restart|status}"
        exit 1
esac

exit 0
```

Save and close the file.

4. Change the file to executable, change ownership, install, and start the service:

```
sudo chmod +x /etc/init.d/kafka

sudo chown root:root /etc/init.d/kafka

sudo update-rc.d kafka defaults

sudo service kafka start

sudo service kafka status
```

Create a Topic

Pick any server to run the following:

1. Use the following command to create a topic named `test`:

```
./bin/kafka-topics.sh --zookeeper zookeeper1:2181/kafka --create
--topic test --replication-factor 1 --partitions 3
```

2. Use the following command to describe the topic:

```
./bin/kafka-topics.sh --zookeeper zookeeper1:2181/kafka --topic
test --describe
```

Conclusion

Congratulations — you've completed this hands-on lab!

Tools

Lab Diagram Instant Terminal

Credentials

How do I connect?

Cloud Server Broker 1

Username

cloud_user

Password

Fpe17w))

Broker 1 Private IP

10.0.1.101

Broker 1 Public IP

50.19.68.74

Launch Instant Terminal

How do I connect?

Additional Resources

The company you work for would like to start using Apache Kafka for real-time analytics of their application data. They realize the benefit to Kafka, in that it can process millions of messages in real-time. The company would like to start using Kafka as soon as possible and have tasked you with building the Kafka cluster. You will be responsible for building a Kafka cluster with three brokers, and three Zookeeper servers. Your company only gave you three servers total, and would like the cluster configured for high-availability and fault-tolerance. Your boss says that if one of the servers fail, we must ensure that the cluster still remains up and running. That means you must include at least two replicas of the data. Here are the steps you must take to build the Kafka cluster:

1. Download the latest Kafka binaries
2. Install Java
3. Disable RAM swap
4. Create a directory with appropriate permissions for Kafka logs
5. Create a directory with appropriate permissions for Zookeeper snapshots
6. Specify an ID for the Zookeeper servers to reach quorum
7. Modify the Kafka configuration file to reflect the appropriate settings
8. Modify the Zookeeper configuration file to reflect the appropriate settings
9. Create Zookeeper as a service, so it can run without manual intervention
10. Create Kafka as a service, so it can run without manual intervention
11. Create a topic to test that the cluster is working properly

Learning Objectives

0 of 7 completed

☐ **Download the Kafka Binaries**

1. Use `wget` to download the tar file from the mirror:

   ```
   wget
   http://mirror.cogentco.com/pub/apache/kafka/2.6.2/kafka_2.
   2.6.2.tgz
   ```

2. Extract the tar file and move it into the `/opt` directory:

   ```
   tar -xvf kafka_2.12-2.6.2.tgz
   ```

   ```
   sudo mv kafka_2.12-2.6.2 /opt/kafka
   ```

3. Change to the `kafka` directory and list the contents:

   ```
   cd /opt/kafka
   ```

   ```
   ls
   ```

☐ **Install Java and Disable RAM Swap**

1. Use the following command to install the latest Java Developer Kit (JDK):

   ```
   sudo apt install -y default-jdk
   ```

2. Use the following command to verify that Java has been installed:

   ```
   java -version
   ```

3. Use the following command to disable RAM swap:

   ```
   swapoff -a
   ```

4. Use the following command to comment out swap in the `/etc/fstab` file:

   ```
   sudo sed -i '/ swap / s/^/#/' /etc/fstab
   ```

☐ **Create a New Directory for Kafka and Zookeeper**

1. Use the following command to create a new directory for Kafka message logs:

   ```
   sudo mkdir -p /data/kafka
   ```

2. Use the following command to create a snapshot directory for Zookeeper:

   ```
   sudo mkdir -p /data/zookeeper
   ```

3. Change ownership of those directories to allow the user `cloud_user` control:

```
sudo chown -R cloud_user:cloud_user /data/kafka

sudo chown -R cloud_user:cloud_user /data/zookeeper
```

☐ Specify an ID for Each Zookeeper Server

1. Use the following command to create a file in `/data/zookeeper` (on server #1) called `myid` with the contents "1" to specify Zookeeper server #1:

```
echo "1" > /data/zookeeper/myid
```

2. Use the following command to create a file in `/data/zookeeper` (on server #2) called `myid` with the contents "2" to specify Zookeeper server #2:

```
echo "2" > /data/zookeeper/myid
```

3. Use the following command to create a file in `/data/zookeeper` (on server #3) called `myid` with the contents "3" to specify Zookeeper server #3:

```
echo "3" > /data/zookeeper/myid
```

☐ Modify the Kafka and Zookeeper Configuration Files

1. Use the following command to remove the existing `server.properties` file (in the `config` directory) and create a new `server.properties` file:

```
rm config/server.properties

vim config/server.properties
```

2. Copy and paste the following into the contents of the `server.properties` file and change the `broker.id` and the `advertised.listeners`:

```
# change this for each broker
broker.id=[broker_number]
# change this to the hostname of each broker
advertised.listeners=PLAINTEXT://[hostname]:9092
# The ability to delete topics
delete.topic.enable=true
# Where logs are stored
log.dirs=/data/kafka
# default number of partitions
num.partitions=8
# default replica count based on the number of brokers
default.replication.factor=3
```

```
# to protect yourself against broker failure
min.insync.replicas=2
# logs will be deleted after how many hours
log.retention.hours=168
# size of the log files
log.segment.bytes=1073741824
# check to see if any data needs to be deleted
log.retention.check.interval.ms=300000
# location of all zookeeper instances and kafka directory
zookeeper.connect=zookeeper1:2181,zookeeper2:2181,zookeepe

# timeout for connecting with zookeeper
zookeeper.connection.timeout.ms=6000
# automatically create topics
auto.create.topics.enable=true
```

3. Use the following command to remove the existing `zookeeper.properties` file (in the `config` directory) and create a new `zookeeper.properties` file:

   ```
   rm config/zookeeper.properties
   ```

   ```
   vim config/zookeeper.properties
   ```

4. Copy and paste the following into the contents of the `zookeeper.properties` file (don't change this file):

   ```
   # the directory where the snapshot is stored.
   dataDir=/data/zookeeper
   # the port at which the clients will connect
   clientPort=2181
   # setting number of connections to unlimited
   maxClientCnxns=0
   # keeps a heartbeat of zookeeper in milliseconds
   tickTime=2000
   # time for initial synchronization
   initLimit=10
   # how many ticks can pass before timeout
   syncLimit=5
   # define servers ip and internal ports to zookeeper
   server.1=zookeeper1:2888:3888
   server.2=zookeeper2:2888:3888
   server.3=zookeeper3:2888:3888
   ```

- [ ] **Create the Kafka and Zookeeper Service**

1. Create the file `/etc/init.d/zookeeper` on each server and paste in the following contents:

```
sudo vim /etc/init.d/zookeeper

#!/bin/bash
#/etc/init.d/zookeeper
DAEMON_PATH=/opt/kafka/bin
DAEMON_NAME=zookeeper
# Check that networking is up.
#[ ${NETWORKING} = "no" ] && exit 0

PATH=$PATH:$DAEMON_PATH

case "$1" in
  start)
        # Start daemon.
        pid=`ps ax | grep -i 'org.apache.zookeeper' |
grep -v grep | awk '{print $1}'`
        if [ -n "$pid" ]
          then
            echo "Zookeeper is already running";
        else
          echo "Starting $DAEMON_NAME";
          $DAEMON_PATH/zookeeper-server-start.sh -daemon
/opt/kafka/config/zookeeper.properties
        fi
        ;;
  stop)
        echo "Shutting down $DAEMON_NAME";
        $DAEMON_PATH/zookeeper-server-stop.sh
        ;;
  restart)
        $0 stop
        sleep 2
        $0 start
        ;;
  status)
        pid=`ps ax | grep -i 'org.apache.zookeeper' |
grep -v grep | awk '{print $1}'`
        if [ -n "$pid" ]
          then
```

```
            echo "Zookeeper is Running as PID: $pid"
          else
            echo "Zookeeper is not Running"
          fi
          ;;
    *)
          echo "Usage: $0 {start|stop|restart|status}"
          exit 1
esac

exit 0
```

2. Change the file to executable, change ownership, install, and start the service:

```
sudo chmod +x /etc/init.d/zookeeper

sudo chown root:root /etc/init.d/zookeeper

sudo update-rc.d zookeeper defaults

sudo service zookeeper start

sudo service zookeeper status
```

3. Create the file `/etc/init.d/kafka` on each server and paste in the following contents:

```
#!/bin/bash
#/etc/init.d/kafka
DAEMON_PATH=/opt/kafka/bin
DAEMON_NAME=kafka
# Check that networking is up.
#[ ${NETWORKING} = "no" ] && exit 0

PATH=$PATH:$DAEMON_PATH

# See how we were called.
case "$1" in
  start)
        # Start daemon.
        pid=`ps ax | grep -i 'kafka.Kafka' | grep -v grep
| awk '{print $1}'`
        if [ -n "$pid" ]
          then
```

```
            echo "Kafka is already running"
        else
          echo "Starting $DAEMON_NAME"
          $DAEMON_PATH/kafka-server-start.sh -daemon
/opt/kafka/config/server.properties
        fi
        ;;
  stop)
        echo "Shutting down $DAEMON_NAME"
        $DAEMON_PATH/kafka-server-stop.sh
        ;;
  restart)
        $0 stop
        sleep 2
        $0 start
        ;;
  status)
        pid=`ps ax | grep -i 'kafka.Kafka' | grep -v grep
| awk '{print $1}'`
        if [ -n "$pid" ]
          then
          echo "Kafka is Running as PID: $pid"
        else
          echo "Kafka is not Running"
        fi
        ;;
  *)
        echo "Usage: $0 {start|stop|restart|status}"
        exit 1
esac

exit 0
```

Save and close the file.

4. Change the file to executable, change ownership, install, and start the service:

```
sudo chmod +x /etc/init.d/kafka

sudo chown root:root /etc/init.d/kafka

sudo update-rc.d kafka defaults

sudo service kafka start
sudo service kafka status
```

☐ Create a Topic

1. Use the following command to create a topic named `test`:

   ```
   ./bin/kafka-topics.sh --zookeeper zookeeper1:2181/kafka --create --topic test --replication-factor 1 --partitions 3
   ```

2. Use the following command to describe the topic:

   ```
   ./bin/kafka-topics.sh --zookeeper zookeeper1:2181/kafka --topic test --describe
   ```
